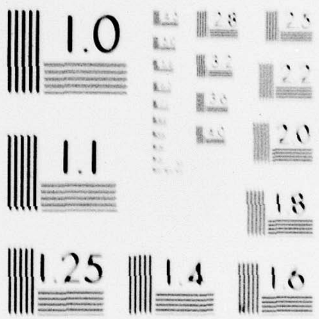
AD-A075 095 NAVAL RESEARCH LAB WASHINGTON DC F/G 9/2
EXPAND: AN EDITING ROUTINE FOR STROMBERG-CARLSON MOVIE FILES.(U)
SEP 79 R C SHOCKLEY , M J FRITTS
UNCLASSIFIED NRL-MR-4061 NL

1 OF 1
AD
A075095

END
DATE
FILMED
3-80
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

NRL Memorandum Report 4061

Expand: An Editing Routine for Stromberg-Carlson Movie Files

R. C. SHOCKLEY AND M. J. FRITTS

Laboratory for Computational Physics

September 20, 1979

NAVAL RESEARCH LABORATORY
Washington, D.C.

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Memorandum Report 4061	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) EXPAND: AN EDITING ROUTINE FOR STROMBERG-CARLSON MOVIE FILES		5. TYPE OF REPORT & PERIOD COVERED Continuing 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) R.C. Shockley and M.J. Fritts		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, D.C.		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Proj. No. RR 014-03-02 NRL J.O. 62H02-39
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research 800 N. Quincy St. Arlington, Va 22203		12. REPORT DATE September 20, 1979 13. NUMBER OF PAGES 22
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This work was supported by the National Research Council and the Office of Naval Research under Project No. RR014-03-02

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Graphics	Subroutines
Movie Files	Color Movies
Editing	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The increasing use of movies in various research activities at NRL has created a demand for subroutine packages which make and edit movie files which contain Stromberg-Carlson commands. EXPAND and EDITSC are respectively, a worker subroutine and a simplified FORTRAN driver subroutine which allow considerable freedom in editing movie files. Options include replicating or deleting arbitrary frames or groups of frames, inserting blank frames between any frames or group of frames (to give borders for handmade slides), merging separate movie files, and splitting

(Continued)

20. ABSTRACT (Continued)

one movie into several. Details of these routines are discussed in sufficient depth to indicate where changes in EXPAND could be made to increase the editing flexibility (e.g. the addition or changing of color wheel commands, or circumventing the limits in EXPAND on the size of input and output files).

CONTENTS

Expand: An Editing Routine For Stromberg-Carlson Movie Files

I. INTRODUCTION

A rapidly growing number of research activities across a broad range of scientific fields use computer-generated movies. Movies provide a unique type of physical insight, that of time-evolution, into problems of all types, and invaluable intuition in analyzing *transient* phenomena in, for example, fluid flows, combustion processes, and electromagnetic and quantum mechanical wave packet propagation. In addition, movies often shed light on purely numerical problems. They are unrivaled in efficiency of communication, and provide their makers with a satisfying, elegant format.

Nevertheless, movie-making usually burdens computational scientists with extra debugging requirements and larger storage requirements. Even when the data-generating routines are debugged, a movie file when first produced will seldom represent the best film possible; titles, summaries, or breaks between "scenes" may be left out or the rate of time evolution (at typical projector speeds) may be unacceptable. Because a code that generates movie data typically is long and complex, the job of tailoring it to allow flexibility in movie making is doubly frustrating for workers desiring a quick presentation of numerical results.

The increasing use of movies in research and the need to simplify movie-making have created a strong interest at NRL in specialized subroutines which modify Stromberg-Carlson

Manuscript submitted June 26, 1979.

(SC) command movie files. (See GRAFIT. Ref. 1.) This report describes EXPAND, a routine for editing SC files, which should alleviate several of the problems presently found in moviemaking.

In short, EXPAND permits one to operate on the file exactly as if it were a reel of film (except EXPAND does not scan backwards). EXPAND as presently written has options for deleting selected frames (singly or in groups), replicating frames so as to slow down or stop the apparent passage of time, merging or splitting movie files, and inserting blank frames between frames with data to facilitate slide preparation.

A set of examples are provided showing typical jobs using EXPAND in the last section (V) of this report. The preceding sections deal with the action of EXPAND, the calling sequence, the JSL (Job Specification Language), and desirable changes in the present source code to add options. Additional options might include a color-wheel-change mode, or a set of routines to detect and segment any extremely long output file requests by scanning the EXPAND input data before generating the output file, and initiating appropriate FOT (file output to tape) commands.

II. GENERAL DESCRIPTION OF EXPAND AND EDITSC

EXPAND requires for input one Stromberg-Carlson (SC) file, which must be named FTn1F001 where n1 is a two-digit positive integer. It creates a second file, FTn2F001. Other SC files will be denoted similarly. The user must supply n1, n2, etc., and provide appropriate FIT, FOT and FD commands (see Section IV).

Reading input files from disks and writing output files to disks is done with READFL and WRITFL (see Ref. 2), but EXPAND sets up all necessary parameters to use READFL and

WRITFL. Frames are counted in the input file by searching each input block for an SC "advance-frame" command.

For most applications of EXPAND, the desired result will be a new file with each frame replicated some number of times. (See Fig. 1a.) This has the effect of slowing the time evolution by the repetition factor, but also note that one can stop time by repeating a single frame many times. Thus EXPAND facilitates the use of special *still* frames, which need not be generated during the initial movie-making run, with special printed matter, annoted figures, or similar information. Note that input color movie files must begin each frame with the color wheel on 'clear'. Otherwise replication of frames will sequence the colors in an unpredictable manner.

Figure 1 shows the effect of other types of editing options, although only for the simplest types of applications. A more general output film could be made with EXPAND in which part of the input film has been expanded, another part deleted, another part spaced by blank frames, and a fourth part has been added from a different input film, for example.

Although conceptually simple, the logic involved in a general input and output, frame-counting algorithm is complicated by the variable length of frames and the block nature of data transmission to and from tapes. In our case the blocksize is 2880 bytes (720 single precision words). A more fundamental source of difficulty is the impracticality of bringing an entire movie file into core. Hence we discuss EXPAND in this section as if it operated on a single reel of film, without regard for physical data flow.

Nevertheless, an important limit which results from this block nature should be pointed out: A single frame in FTn1F001 must not exceed the equivalent of 7220 32 bit words. A

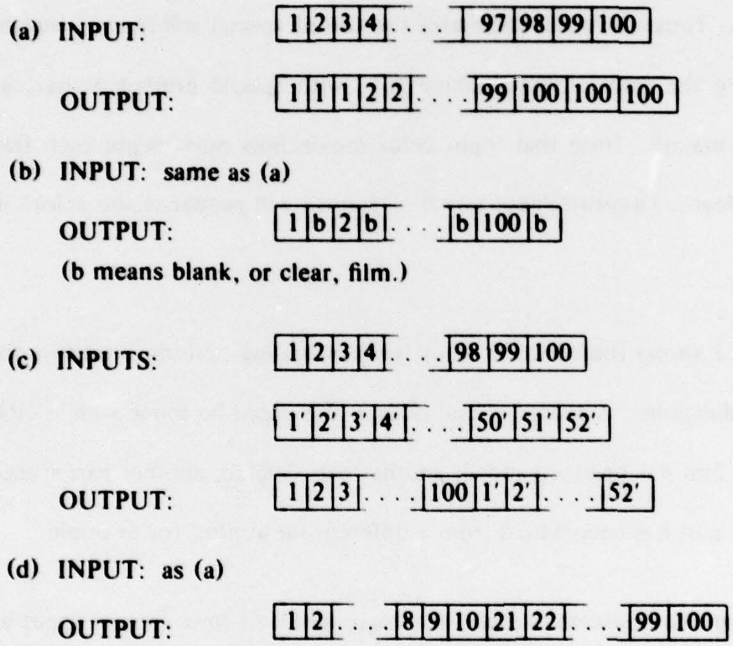


Figure 1. A number of editing options are available using EXPAND. Indicated schematically in (a) is simple replication of frames, while (b) shows the insertion of blanks, (c) shows merging, (d) shows deletion.

longer set of commands per frame causes EXPAND to miscount frames, or terminate abnormally, since no more than ten blocks are allowed from FTn1F001 in core in the search for a new advance command. For most applications, this limit is liberal. It may be raised by simple changes in the source code. Other limits in the present EXPAND will be ennumerated after we have described the calling sequence in Section III.

Multiple calls to EXPAND or EDITSC (see Section III) may occur from a single program. Because the same output file (n2) can be used repeatedly when calling EXPAND (but not EDITSC), one must include the data (MODE, in Section III) to keep it open or to close it after each call. EDITSC automatically generates the right data for this, by reading the next card after its present instructions, and then calling EXPAND.

The penalty for using EDITSC, which in general is more convenient than EXPAND, is that one call processes a *contiguous* segment of the film (physically, the SC file), and each frame of this segment is treated identically. With EXPAND, one call may invoke a *different* action on each frame or groups of contiguous frames.

Any attempt to *reread* sections of the film which have been passed will fail unless the input file I1 is first closed. This occurs only if another call to EXPAND (or EDITSC) references a different input file. In other words, successive calls to EXPAND using the same input film cannot change the chronological order on the output film. Having closed n1, EXPAND will automatically begin counting frames again, on the next call using n1, calling the first frame one.

For clarity we note here that when successive calls to EXPAND or EDITSC ask for the same input file, the frames on each call are counted starting with number one as the next unprocessed frame after the previous call. As the next section will show, only frame advances are counted and no record is kept of the absolute frame number for each file.

III. CALLING SEQUENCE

This section defines the parameters used in calling EXPAND and EDITSC. Remarks on JSL and program limits are discussed in Section IV. We describe EDITSC first since it is somewhat simpler than EXPAND and will suffice for many editing purposes. The method for calling EXPAND, which allows more general options, is very similar to that of EDITSC. Both are on a library at node USERCAT/LIB/LCP/SHOCR1/EXPAND, containing source (../EXPAND/SOURCE) and object (.../EXPAND/OBJECT) modules so compilation is unnecessary, and those interested may obtain source listings.

A. CALLING EDITSC

EDITSC is called with the statement

CALL EDITSC,

which causes *data cards* to be read. Each card is assumed to contain the variables:

N1 N2 STF ENF REP

under the format 2I3, 2I5, I3. These integer variables are defined as follows:

N1 = input file number (in FTn1F001),

N2 = output file number (in FTn2F001),

STF = starting frame number on N1 at which to begin copying to N2,

ENF = ending frame number on N1, and

REP = number of replicas of each frame of N1 to be copied to N2.

As mentioned above, N1 and N2 must be positive two-digit integers, that is, exceeding 9 but not 99. There is one exception: one may terminate the set of cards with the datum N1 = -1 and any other values (or blanks) for the remaining variables. When EDITSC finds that N1 $\leq$ 9 on any card except the first, it returns control to the calling program automatically, after processing the preceeding cards. If it finds N1 $\leq$ 9 on the first card, it returns control and prints an error message.

Briefly, each card causes ENF-STF+1 frames to be copied, each REP times, from N1 to N2. If blank frames are desired between each frame on N2, one simply makes the sign of REP negative. For the data cards ("b" denotes blank):

```
/bXQT
b10b20bbb10bbb90b-3
b-1
/bEOJ
```

EDITSC should copy 81 frames from FT10F001 to FT20F001 (see Fig. 2(a)), replicating each frame three times, and inserting blank frames beginning with frame 10 and ending with frame 90. (As discussed in Section IV, the user must provide appropriate FIT, FOT, and FD statements for all files used.)

A data set using EDITSC to break a movie into two separate movies, one at half the present time evolution rate and a second with blank frames between each frame (see Fig. 2(b)), might appear as

```
/bXQT

b15b21bbbb1bb150bb2

b15b22bb151bb200b-1

b-1

/bEOJ
```

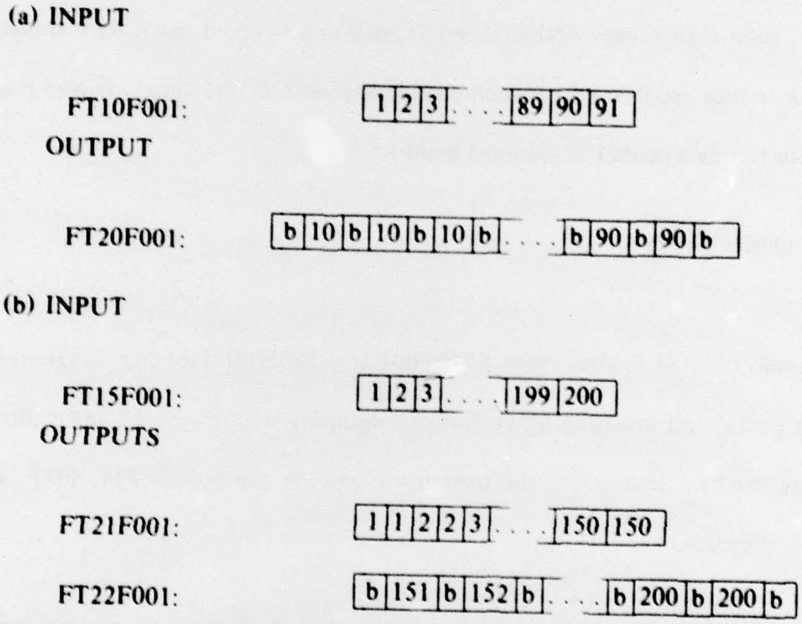


Figure 2. Schematic illustration of the action caused by data cards shown in the text. Note the extra blank frames before the first and after the last data frame when the blank-frame option is used.

B. CALLING EXPAND

EXPAND is called with the statement

CALL EXPAND (N1, N2, FRAME, REPS, NFRAMS, MODE),

which, in contrast to EDITSC, does not cause cards to be read, but otherwise performs similar operations. The arguments, all of type integer, are:

N1 = input file number (in FTn1F001),

N2 = output file number (in FTn2F001),

FRAME = the FORTRAN name of a one dimensional array of length 10,000, whose i*th* element is the number of the i*th* frame in FTn1f001 to be processed.

REPS = the FORTRAN name of a one dimensional array of length 10,000, whose i*th* element is the number of replicas of the i*th* frame (FRAME (I)) on FTn1F001 to be passed to FTn2F001,

NFRAMS = the total number of frames to be processed on FTn1F001, and

MODE = an integer control variable with the effect:

0, close N2 on return, no blank frames,

1, close N2 on return, insert blank frames,

2, do not close N2 on return, no blank frames,

and

3, do not close N2 on return, insert blank frames.

The same limits apply to N1 and N2 as in EDITSC ($9<N1, N2 < 100$). The array FRAME must contain an increasing series.

Note that EXPAND allows one freedom to replicate certain frames more than others. EDITSC corresponds to a subset of EXPAND calls with all elements of array REP equal. EXPAND does not, however, allow different blank-frame insertion options for different frames. The generalization of MODE to allow this requires straightforward changes in the code, but the benefits to be gained are not clear; it would require the user's code to generate another more complicated array before calling EXPAND, whereas the same effect is available with multiple calls.

EXPAND also permits arbitrary frames to be deleted, through the FRAME array. In order to pick up only odd numbered frames, one would define FRAME before calling EXPAND as

FRAME (i) = 2i-1,

for example.

IV. JSL REMARKS AND PROGRAM LIMITS

This section contains remarks on the necessary JSL to make runs with EXPAND, and a list of the present restrictions on various parameters in EXPAND. A number of the following remarks have appeared in preceding sections.

A. JSL Remarks

JSL needed to run EXPAND includes FD (File Declaration), FIT (File Input from Tape), FOT (File Output to Tape), and MFR-MFRE (Multiple File Request, Multiple File Release) groups. They will be discussed in Section V.

In order to match the READFL and WRITFL routines, input and output files should have the following characteristics:

name: FTnnF001, (9<nn<100)

FORG (file organization): PS (physical sequential),

BKSZ (blocksize): 2880 (in bytes),

LREC (record length): 2880 (in bytes), and

RCFM (record format): FB (fixed block).

The FORG, BKSZ, LREC and RCFM parameters are all initialized properly in the parameter files used by WRITFL, and need not appear on an FD card for output files. Hence a typical FD card would appear as

```
/bFDbFT10F001, BAND = 10/15/5, FORG = PS, BKSZ = 2880,;

bbbLREC = 2880, DTYP = PAD, RCFM = FB
```

FIT's and FOT's move the movie files between disks and tapes. Stomberg-Carlson commands are stored on tapes with characteristics

DEN (density) = 556 (bits per inch),

TRKS (tracks) = 7,

RCTL (recording control) = C (odd parity with conversion but no translation), and

LABL (label) = 1/NL (unlabeled tape).

Typical FIT and FOT commands might read

```
/bFITbFTn1F001, DEN = 556, TRKS = 7, RCTL = C,;

bbbLABL = 1/NL, EFID = 500619
```

and

```
/bFOT FTn2F001, DEN = 556, TRKS = 7, RCTL = C,;

bbbLABL = 1/NL, EFID = 500414
```

Here n1 are n2 should be the same file identifiers appearing in calls to EDITSC and EXPAND.

As mentioned earlier, the routines EDITSC and EXPAND are stored in a library at USERCAT/LIB/LCP/SHOCR1/EXPAND. A library card in the linkage step should appear, as in:

```
/bLNK

LIBRARY DUMMY
```

where DUMMY is an arbitrary access name previously assigned to the node where EXPAND is located:

```
/bASGbDUMMY, USERCAT/LIB/LCP/SHOCR1/EXPAND, USE=SHR
```

For jobs with several files, the ASC default values for ADDMEM may be too small. In this case, the message

```
. . . ABEND GM01 . . .
```

will appear in the job activity file. To compute the required ADDMEM for a given file, use the expression (See Reference 3):

80 + (number of words in LREC rounded up to a multiple of 8)
+ (number of words in BKSZ rounded up to a multiple of 8).

For the SC files commonly used with LREC = 2880 and BKSZ = 2880 (in bytes), this formula gives 80 + 720 + 720 = 1520 words of ADDMEM per file. Additional ADDMEM is indicated on the /XQT card:

/bXQTbADDMEM = 7K

B. Program Limits

The present verion of EXPAND (or EDITSC where appropriate) requires that :

1) file designators n1, n2, satisfy 9<nn<100;

2) each frame on an input file must is less than or equal to 7200 ASC words (this is ten 2880-byte blocks);

3) the total number of frames to be copied from FTn1F001 (ENF-STF+1 for EDITSC, NFRAMS for EXPAND) does not exceed 10,000 (ignoring repetitions);

4) The maximum number of replicas of any frame (absolute value of REP for EDITSC or of REP (I) for EXPAND) is 99;

5) EDITSC is expected to read no more than 500 data cards, including the flag for normal termination.

6) files are short enough to fit on available PAD (currently about 1500 BANDS).

V. EXAMPLES

Section V.A. presents examples of complete decks for using EXPAND. These examples serve only as guides, not tutorial references, for setting up the JSL statements. They show typical methods for file handling, and assume knowledge of ASC JSL. MFR-MFRE groups are also illustrated. In Section VB we shall discuss printed output briefly.

A. Examples of ASC Jobs

Since EXPAND requires the user to generate editing-instruction arrays, but does not affect the JSL differently from EDITSC in any way, the examples only show calls to EDITSC. Data cards therefore are shown after the XQT statement, which would not be required if EXPAND were used. Numbers and names used in this section for disk space reservation, time limits, file sizes, data, path names, and file names are purely for illustrative purposes.

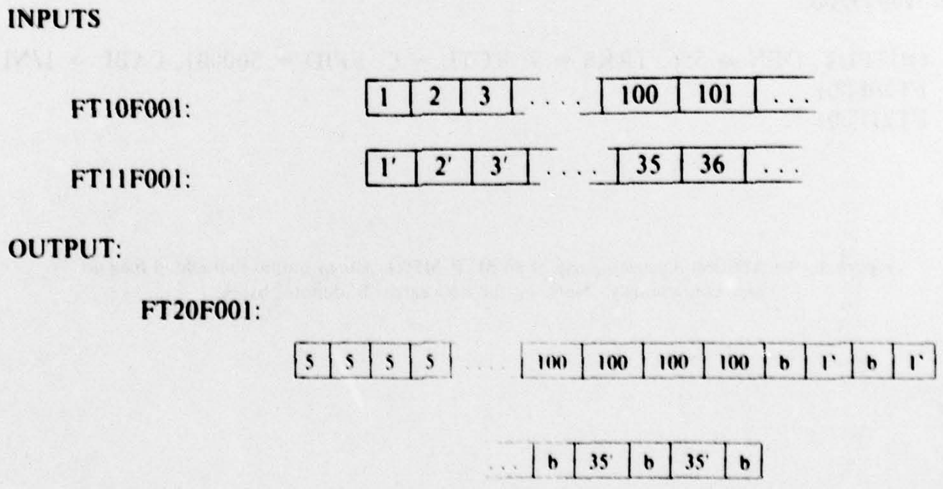
The job in Figure 3 makes use of EDITSC to edit two input movie files stored on the tapes with EFID's 500835 and 500901 and merge the resulting file on a third file which is stored on the tape with EFID = 500855. (Users must have registered these tapes previously, and they must be available at run time to the ASC.)

The results of this deck are that frames 5-100 of FT10F001 are copied each four times onto FT20F001, followed by frames 1-35 from FT11F001 which are copied each twice and separated by blanks as indicated by Figure 4. The −1 on the third data card causes normal return to the dummy calling program.

Figure 5 presents a more complicated case in which two input files (not on tape) are expanded by a factor of two (halving the apparent rate of time evolution for a given projector

```
/ JOB     EXAMPL1, 7777777, USERCD, OPT = (C,R)
/ LIMIT BAND = 100, BAND = 700/PAD, MIN = 1
/ ASG     EDITLIB, USERCAT/LIB/LCP/SHOCR1/EXPAND/OBJECT, USE = SHR
/ FD      FT06F001, BAND = 5/40/5
/ FD      FT10F001, BAND = 10/15/5, FORG = PS, BKSZ = 2880, LREC = 2880, DTYP = PAD,;
              RCFM = FB
/ FD      FT11F001, BAND = 10/15/5, FORG = PS, BKSZ = 2880, LREC = 2880, DTYP = PAD,;
              RCFM = FB
/ FD      FT20F001, BAND = 20/30/5, DTYP = PAD
/ FIT     FT10F001, DEN = 556, TRKS = 7, RCTL = C, LABL = 1/NL, EFID = 500835
/ FIT     FT11F001, DEN = 556, TRKS = 7, RCTL = C, LABL = 1/NL, EFID = 500901
/ FTN
C
C THIS IS A DUMMY PROGRAM TO CALL EXPAND, VIA EDITSC
C
        CALL EDITSC
        STOP
        END
/ LNK
 LIBRARY EDITLIB
/ FXQT  ADDMEM = 7K
b10b20bbbb5bb100bb4
b11b20bbbb1bbb35b-2
b-1
/ FOT     FT20F001, DEN = 556, TRKS = 7, RCTL = C, LABL = 1/NL/ADD, EFID = 500855
/ EOJ
(7/9 card)
```

Figure 3. An ASC job illustrating a simple use of EDITSC. Note : "b" means a blank on the data cards.



INPUTS

FT10F001: | 1 | 2 | 3 | .... | 100 | 101 | ... |

FT11F001: | 1' | 2' | 3' | .... | 35 | 36 | ... |

OUTPUT:

FT20F001:

| 5 | 5 | 5 | 5 | .... | 100 | 100 | 100 | 100 | b | 1' | b | 1' |

... | b | 35' | b | 35' | b |

Figure 4. The effect of the deck shown in Figure 3.

```
/ JOB    EXAMPL2, 7777777, USERCD, OPT = (C,R)
/ LIMIT  BAND = 100, BAND = 700/PAD, MIN = 2
/ FD     FT06F001, BAND = 5/40/5
/ PD     WAMOVY, USERCAT/D74/B30/USERCD/SPLISH/STROMB
/ FD     FT11F001, BAND = 10/50/5, FORG = PS, BKSZ = 2880, LREC = 2880, RCFM = FB,;
         DTYP = PAD
/ ASG    FT11F001, WAVMOVY/NO350, USE=SHR
/ FD     FT12F001, BAND = 10/50/5, FORG=PS, BKSZ = 2880, LREC = 2880, RCFM = FB,;
         DTYP = PAD
/ ASG    FT12F001, WAVMOVY/NO352, USE = SHR
/ FD     FT20F001, BAND = 50/250/50, DTYP = PAD
/ FD     FT21F001, BAND = 50/250/50, DTYP = PAD
/ ASG    EDITLIB,USERCAT/LIB/LCP/SHOCR1/EXPAND/OBJECT, USE = SHR
/ FTN
C
C  THIS IS ANOTHER DUMMY PROGRAM TO CALL EXPAND
C
        CALL EDITSC
        STOP
        END
/ LNK
bLIBRARY EDITLIB
/ FXQT  CPTIME = 12000, ADDMEM = 14K
b11b20bbbb1bb295bb2
b12b21bbbb1bb295bb2
b-1
/ MFR    OUTPUT, DEN = 556, TRKS = 7, RCTL = C, EFID = 500001, LABL = 1/NL/ADD
/ FOT    FT20F001
/ FOT    FT21F001
/ MFRE
/ EOJ
(7/9 card)
```

Figure 5. An ASC job illustrating use of an MFR-MFRE pair to output two edited files on tape contiguously. Note: on the data cards "b" denotes blank.

speed), placed on separate files, and output to a tape. The MFR-MFRE pair illustrates how files being output to tape (FOT'd) may be stored contiguously.

A third example shown in Fig. 6, works with six input files initially stored as contiguous, pairs on three tapes. MFR-MFRE pairs and used in this example for input (FIT commands) as well as for output (FOT commands). Output files are copied onto a single tape.

B. Remarks on Printed Output

EXPAND and EDITSC have a number of self-explanatory, printed output comments. EDITSC prints each card it reads. Two cards are read before EXPAND may be called (to find out what MODE value is appropriate by knowing in advance the next output file number). EXPAND prints out the names of the two files currently being used for input and output, and all other input data from EDITSC, namely NFRAMS, MODE, and the first NFRAMS elements of the arrays FRAME and REPS (See Section III.B).

EXPAND lists every frame of interest as it is found, together with the number of calls to WRITFL required to copy that frame from BBUF (the array in core which holds each frame) to the output (disk) file, and the number of calls made to WRITFL.

EDITSC will catch and report improperly formatted input cards (e.g. any card with a real number in the first thirteen columns), and an inconsistent frame request with the starting frame farther along the film than the ending frame (STF > ENF; see Section III.A).

EXPAND will print an error message, different in each case, and terminate normally if any of the following happen:

```
/     JOB        EXAMPL3, 7777777, USERCD, OPT = (C,R)
/     LIMIT      BAND = 100, BAND = 700/PAD, MIN = 2
/     FD         FTO6FOO1, BAND = 5/40/5
/     FD         FT10F001, BAND = 10/150/5, FORG = PS, BKSZ = 2880, LREC = 2880, RCFM = FB,;
                 DTYP = PAD
/     FD         FT11F001, . . .
/     FD         FT12F001, . . .
/     FD         FT13F001, . . .
/     FD         FT14F001, . . .
/     FD         FT15F001, . . .
/     MFR        FIRST, DEN = 556, TRKS = 7, RCTL = C, LABL = 1/NL,;
                 EFID = 500811
/     FIT        FT10FOO1, LABL = 1/NL
/     FIT        FT13F001, LABL = 2/NL
/     MFRE
/     MFR        SECOND, DEN = 556, TRKS = 7, RCTL = C, LABL = 1/NL,;
                 EFID = 500813
/     FIT        FT11FOO1, LABL = 1/NL
/     FIT        FT14F001, LABL = 2/NL
/     MFRE
/     MFR        THIRD, DEN = 556, TRKS = 7, RCTL = C LABL = 1/NL,;
                 EFID = 500814
/     FIT        FT12F001, LABL = 1/NL
/     FIT        FT15F001, LABL = 2/NL
/     MFRE
/     ASG        EDITLIB, USERCAT/LIB/LCP/SHOCR1/EXPAND/OBJECT, USE = SHR
/     FD         FT20 F001, BAND = 50/250/50, DTYP = PAD
/     FD         FT21F001, . . .
/     FD         FT22F001, . . .
/     FTN
C
C     THIS A THIRD DUMMY CODE TO CALL EXPAND
C
         CALL EDITSC
         STOP
         END
/     LNK
      LIBRARY EDITLIB
/     FXQT       CPTIME = 12000, ADDMEM = 14K
b10b20bbbb1bb102bb4
b11b20bbbb3bbb51bb4
b12b20bbbb3bbb28bb2
b13b21bbbb1bb102bb2
b14b21bbbb3bbb51bb2
b15b21bbbb3bbb28bb2
b10b22bbbb3bbbb3b-1
b10b22bb102bb102b-1
b11b22bbb51bbb51b-1
b12b22bbb20bbb28b-1
b13b22bb102bb102b-1
b14b22bbb51bbb51b-1
b15b22bbb20bbb28b-1
b-1
/     MFR    OUTPUT, DEN = 556, TRKS = 7, RCTL = C, EFID = 500968,;
             LABL = 1/NL/ADD
/     FOT    FT20F001
/     FOT    FT21F001
/     FOT    FT22F001
/     MFRE
/     EOJ
(7/9 card)
```

Figure 6. An ASC job illustrating the use of MFR-MFRE pairs for FIT and FOT. The data cards select single frames in FT10F001, FT11F001, FT13F001, and FT14F001 for transfer to FT22F001 with the blankframe option. Such multiple calls produce only one blank between each data frame, not two.

(1) NFRAMS is negative,

(2) NFRAMS is greater than 10,000,

(3) elements in array FRAME are not an increasing series,

(4) READFL has an abnormal return, or

(5) Inconsistent pointers to arrays are generated by EXPAND in its internal frame-counting routine.

If the above diagnostics are insufficient for debugging purposes, one may reset the logical variable DIAG from .FALSE. to .TRUE. in the corresponding data initialization line in EXPAND. This produces an extremely detailed history of file designators n1 and n2 (which are obvious candidates for scrutiny if READFL or WRITFL return with errors), input arrays for READFL and WRITFL, and several internal counters and pointers.

REFERENCES

1. J.P. Boris, E. Dent, M.J. Fritts, I. Haber and W.W. Jones, "GRAFIT; A Generalized Graphics Package,", NRL Memorandum Report 4027, 1979.

2. NRL Computer Note 165, 9 June, 1978. See also Sections I9.000 and J9.000 of the ASC Subprogram Library Notebook.

3. NRL Computer Note 165, 9 June, 1978.